CLAIMS

1. A graph walking system, comprising:

a binding system for binding a graph observer with a graph, for binding node patterns to node observers to generate at least one node pairing, and for binding the graph observer to at least one node pattern-node observer pairing;

graph walking logic for systematically walking through nodes within the directed non-cyclic graph;

a pattern testing system for determining if an encountered node matches one of the node patterns;

an event manager for generating an encountered event when one of the node observers is bound to a matching node pattern; and

a pruning system that can deactivate the graph observer with respect to sub-nodes of the encountered node if a bound node observer determines that there is no interest in the sub-nodes.

2. The graph walking system of claim 1, wherein the encountered event is handled by the bound node observer.

3. The graph walking system of claim 1, wherein the graph walking logic walks through the graph in a top down hierarchal manner.

4. The graph walking system of claim 1, wherein the pruning system can reactivate a deactivated graph observer after the sub-nodes of the encountered node have been walked.

5. The graph walking system of claim 1, wherein the event manager generates a completed event for each node observer that received an encountered event and that did not cause the graph observer to become deactivated.

6. The graph walking system of claim 5, wherein the completed event can cause the graph walking logic to repeat the walk through the sub-nodes.

7. The graph walking system of claim 1, wherein the pruning system can further cause the graph walking logic to bypass walking of the sub-nodes if the graph observer has been deactivated and no other active graph observers exist.

8. A system for analyzing a graph of hierarchical data, comprising:

a system for binding a plurality of graph observers to a graph, wherein each graph observer is further bound to a set of node patterns and a set of node observers;

graph walking logic for systematically walking through nodes within the graph;

a first pruning system that can be instructed by a node observer bound with an associated graph observer to deactivate the associated graph observer until a set of sub-nodes for the encountered node has been walked; and

a second pruning system that can instruct the graph walking logic not to walk the set of sub-nodes for the encountered node.

9. The system of claim 8, wherein the second pruning system will cause the set of sub-nodes not to be walked only if all of the plurality of graph observers have been deactivated.

10. The system of claim 8, further comprising a pattern testing system for determining if the encountered node matches one of the node patterns.

11. The system of claim 8, further comprising an event manager for generating an encountered event when one of the node observers is bound to a matching node pattern.

12. A method for analyzing a graph of hierarchical data, comprising the steps of:

binding a plurality of graph observers to a graph, wherein each graph observer is further bound to a set of node patterns and a set of node observers;

systematically walking through nodes within the graph;

generating an encounter event and handling the encounter event with a bound node observer when one of the node patterns matches an encountered node;

deactivating the graph observer associated with the bound node observer if the bound node observer determines that a set of sub-nodes of the encountered node should be pruned; and

bypassing the walking of the set of sub-nodes if all of the plurality of graph observers have been deactivated.

13. The method of claim 12, comprising the further step of generating a completed event for each node observer that received an encountered event and that did not cause the graph observer to become deactivated.

14. The method of claim 12, comprising the further step of reactivating the graph observer associated with the bound node observer after the set of sub-nodes of the encountered node have been walked.

15. The method of claim 12, comprising the further step of reactivating the graph observer associated with the bound node observer after set of sub-nodes of the encountered node have been bypassed.

16. The method of claim 12, comprising the further step of walking the sub-nodes if at least one graph observer is active.

17. A program product stored on a recordable medium, which when executed, analyzes a graph of hierarchical data, the program product comprising:

program code configured to bind a plurality of graph observers to a graph, wherein each graph observer is further bound to a set of node patterns and a set of node observers;

program code configured to provide graph walking logic for systematically walking through nodes within the graph;

program code configured to provide a first pruning system that can be instructed by a node observer bound with an associated graph observer to deactivate the associated graph observer until a set of sub-nodes for an encountered node has been walked; and

program code configured to provide a second pruning system that can instruct the graph walking logic not to walk the set of sub-nodes for the encountered node.

18. The program product claim 17, wherein the second pruning system will cause the set of sub-nodes not to be walked only if all of the plurality of graph observers have been deactivated.

19. The program product claim 17, further comprising program code configured to provide a pattern testing system for determining if the encountered node matches one of the node patterns.

20. The program product claim 17, further comprising program code configured to provide an event manager for generating an encountered event when one of the node observers is bound to a matching node pattern.